DOI:10.25092/baunfbed. 1347379 *J. BAUN Inst. Sci. Technol., 26(1), 226-236, (2024)*

Coding theory for *h(x)*-Fibonacci polynomials

Öznur ÖZTUNÇ KAYMAK*

Department of Information Technology, Izmir Democracy University, 35140 Karabaglar Izmir, TURKIYE

Geliş Tarihi (Received Date): 21.08.2023
Kabul Tarihi (Accepted Date): 03.12.2023

Abstract

The amount of information transmitted over the internet network has dramatically increased with the prevailing of internet use. As a result of this increase, the algorithms used in data encryption methods have gained importance. In this paper, h(x)-Fibonacci coding/decoding method for h(x)-Fibonacci polynomials is introduced. The proposed method is fast because it is based on basic matrix operations, and it is suitable for cryptographic applications because it uses the ASCII character encoding system. For this reason, it differs from the classical algebraic methods in literature. Furthermore, the fact that h(x) is a polynomial improves the security of cryptography.

Keywords: *Coding/decoding algorithm, h(x)-Fibonacci polynomials, Fibonacci Numbers*

h(x)-Fibonacci polinomları için kodlama teorisi

Öz

İnternet kullanımı gün geçtikçe yaygınlaştıkça, internet ağları üzerinden geçen bilgi miktarı da kayda değer ölçüde artmaktadır. Bu artışın sonucu olarak, bilgi şifreleme metotları da kullanılan algoritmalar da önem kazanmaktadır. Bu çalışmada, h(x)-Fibonacci polinomları için h(x)-Fibonacci şifreleme metodu tanıtılmıştır. Önerilen yöntem, temel matris işlemlerine dayandığından hızlıdır ve ASCII karakter kodlama sistemini kullandığından kriptografik uygulamalara uygundur. Bu nedenle literatürdeki klasik cebirsel yöntemlerden farklılık göstermektedir. Ayrıca h(x)'in bir polinom olması kriptografinin güvenliğini artırır.

Anahtar kelimeler: *Şifreleme/Deşifreleme algoritması, h(x)-Fibonacci polinomları, Fibonacci Sayıları*

* Öznur ÖZTUNÇ KAYMAK, oznur.oztunckaymak@idu.edu.tr, https://orcid.org/0000-0003-3832-9947

1.Introduction and Preliminaries

The Fibonacci polynomials play a significant role in mathematics. [6] These polynomials were first examined by German mathematician E. Jacobsthal and Belgian mathematician Eugene Charles Catalan in 1883. For $n \geq 3$**,** these polynomials $F_n(x)$, $J_n(x)$, $\varphi_n(x)$ are defined

$$F_1(x) = 1, F_2(x) = x, F_n(x) = xF_{n-1}(x) + F_{n-2}(x); \tag{1}$$

$$J_1(x) = J_2(x) = 1, J_n(x) = J_{n-1}(x) + xJ_{n-2}(x); \tag{2}$$

$$\varphi_0(x) = 0, \varphi_1(x) = 1, \varphi_n(x) = 2x\,\varphi_{n-1}(x) + \varphi_{n-2}(x); \tag{3}$$

by Catalan, Jacobsthal, P.F. Byrd, respectively. Fibonacci numbers are obtained by getting $x = 1$ in the equation (1), and these numbers are used in different many applications in the arts, modern sciences and architecture. [6]

The $h(x)$-Fibonacci polynomials, which are the most general form of the polynomials defined the relations (1), (2), (3) above are defined by Nalli and Haukkanen

$$F_{h,n+1}(x) = h(x)F_{h,n}(x) + F_{h,n-1}(x) \tag{4}$$

with initial conditions $F_{h,0}(x) = 0, F_{h,1}(x) = 1$ in [7].

Now, we give the following statements about $h(x)$-Fibonacci polynomials.

Theorem: [7] The characteristic equation for $F_{h,n}(x)$ is given by

$$v^2 - h(x)\,v - 1 = 0$$
(5)

where $\alpha(x) = \frac{h(x)+\sqrt{h^2(x)+4}}{2}$ and $\beta(x) = \frac{h(x)-\sqrt{h^2(x)+4}}{2}$.

Theorem: [7] Let $n \geq 1$. The matrix $\theta_h^n(x)$ is defined as

$$\theta_h^n(x) = \begin{bmatrix} F_{h,n+1}(x) & F_{h,n}(x) \\ F_{h,n}(x) & F_{h,n-1}(x) \end{bmatrix} \tag{6}$$

where $\theta_h\,(x) = \begin{bmatrix} h(x) & 1 \\ 1 & 0 \end{bmatrix}$.

Corollary: [7] The identity $F_{h,n+1}(x)F_{h,n-1}(x) - F_{h,n}^2(x) = (-1)^n$ holds for $n \geq 1$.

Recently, $(p, q) -$ Fibonacci and Lucas polynomials, which are the more general form of $h(x)$-Fibonacci polynomials, have been defined by Aşcı and Lee where $p(\delta)$ and $q(\delta)$ are the real functions in [1]. The general forms of ordinary generating functions of Fibonacci-type numbers and polynomials are given in [2]. Fibonacci–Changhee numbers and polynomials, ${}_{Ch}F_{p,q,n}(\delta)$ are defined by Zhang, Khan, Kızılateş and some features of

these polynomials are obtained in [3]. For other studies of special type polynomials, please see [8-11].
To provide information security of data transfer through communication channel, there have been many algorithms in the literature. Especially, encoding/decoding algorithms have been of crucial role in assuring information security. Particularly, the Fibonacci coding theory stands out as a highly favored approach in this field. For example, in 2006, a new coding theory using the Fibonacci p-numbers and Q_p-matrices is proposed in [13]. Following this study, different methods in this field have been constructed. (see [4,5,12,14,15,16,19,20,25]) There have been studies on Fibonacci and related polynomials' encryption and decryption available in the literature. ([17],[21], [22]) On the other hand, a new cryptography method based on Golden matrices is introduced in [18]. In this study, it has been simply proved that this "Golden" model is a secure and fast cryptographic system.

The main purpose of this study is to present a coding theory of the *h(x)*- Fibonacci polynomials that are a generalization of the matrix [7] for Fibonacci polynomials, Byrid polynomials etc. Also, in this study, the performance of this method is evaluated.

2. A Coding/Decoding method for *h(x)*-Fibonacci polynomials

Here, a coding/decoding method based on *h(x)*-Fibonacci Polynomials is presented. In this method, we insert our message into an even-sized matrix by inserting a zero at the end of the message and every two words up until the matrix size is even. Then, we adapt a coding scheme by splitting the message square matrix *M* of size *2m* into the block matrices termed B_i $(1 < i < m^2)$ of size 2×2.

Note that the following forms for the matrices G_i, B_i and $\theta_h^n(x)$ in (6):

$$G_i = \begin{bmatrix} g_1^i & g_2^i \\ g_3^i & g_4^i \end{bmatrix}, B_i = \begin{bmatrix} b_1^i & b_2^i \\ b_3^i & b_4^i \end{bmatrix}, \text{ and } Q_h^n = \begin{bmatrix} q_1 & q_2 \\ q_3 & q_4 \end{bmatrix} \tag{7}$$

In this method, the approach described in [14] is inspired. But, instead of the character table in [14], we convert all characters to ASCII, which is named (American Standard Code for Information Interchange) character encoding that represents characters as numerical values. In ASCII, each character is assigned a unique code, allowing text to be stored and transmitted as a sequence of numbers. To decode ASCII, you convert the numerical values back into their corresponding characters. Basically, this code is used to recognize characters and numbers on a keyboard. These are eight-bit sequence codes that are used to distinguish between special characters, numbers, and letters. This system is simple to comprehend and uses no complicated steps to execute. [26] Furthermore, because our method is based on polynomials, it differs from this method proposed in [14].

Encryption types can be categorized based on various characteristics. These characteristics are the key used, the algorithms employed and the specific use cases for which they are designed for. There are two common encryption types: Symmetric and asymmetric encryption. Since we use symmetric encryption in this method, we define the public key n as follows:

$$n = \begin{cases} b\,, & b \leq 3 \\ \llbracket b/2 \rrbracket, & b > 3 \end{cases} \tag{8}$$

where b represents the number of the block matrices B_i.

Now, we introduce all the steps of the encryption and decryption of this method:

2.1.Encryption Process:

Step 1: Start.
Step 2: Construct the *M* message matrix.
Step 3: Find n.
Step 4: Convert to string to ASCII decimal.
Step 5: The matrix *M* is subdivided into B_i $(1 < i < m^2)$.
Step 6: Find b_j^i where $(1 \leq j \leq 4)$.
Step 7: Compute det (B_i) and assign to d_i.
Step 8: Build the matrix E.
Step 9: End.

2.2 Decryption Process:

Step 1: Start.
Step 2: Find $Q_h^n(x)$.
Step 3: Assign entries of the $Q_h^n(x)$ matrix to r_j $(1 \leq j \leq 4)$.
Step 4: Find $r_1 b_1^i + r_3 b_2^i \to g_1^i$ $(1 < i < m^2)$.
Step 5: Find $r_2 b_1^i + r_4 b_2^i \to g_2^i$.
Step 6: Solve $(-1)^n \times d_i = g_1^i$ $(r_2 x_i + r_4 b_4^i)$- $g_2^i\left(r_1 x_i + r_3 b_4^i\right)$.
Step 7: Replace by x_i with b_3^i.
Step 8: Find B_i.
Step 9: Construct *M.*
Step 10: Convert *M.*
Step 11: End.

Example 1.1: Assume that we have the subsequent message matrix
Encryption Algorithm:
Now, after starting the algorithm, let us start with the second step:
Step 2: Consider that we have the following message matrix:

$$M = \begin{bmatrix} BELL & LF & ETX & DLE \\ SI & STX & EOT & EOT \\ LF & US & SO & STX \\ NAK & DC3 & LF & EM \end{bmatrix}_{4x4} \tag{9}$$

Note that according to the ASCII table in [23], the equivalent of the *LF(Line Feed)* unicode in the matrix *M* is *newline* [24]. (See another unicode's [23] in the matrix *M*) In this table, we only used *the control code chart table* when composing this message. If desired, a message matrix can be created by *the printable characters* [24].

Step 3: Now, in this step, the public key n is calculated:

$$n = [\![b/_2]\!] = 2$$

using the equation (8).

Step 4: Here, all unicodes in the *M* message matrix in (9) are converted to ASCII decimal in [24]. Firstly, we convert the *M* message matrix as the following form:

$$M = \begin{bmatrix} 7 & 10 & 3 & 16 \\ 15 & 2 & 4 & 4 \\ 10 & 31 & 14 & 2 \\ 21 & 19 & 10 & 25 \end{bmatrix}. \qquad (10)$$

Step 5: The message matrix *M* of size 4×4 in (10) can be divided into the matrices $B_i, 1 \leq j \leq 4$, from left to right, each of size 2×2:

$$B_1 = \begin{bmatrix} 7 & 10 \\ 15 & 2 \end{bmatrix}, B_2 = \begin{bmatrix} 3 & 16 \\ 4 & 4 \end{bmatrix}, B_3 = \begin{bmatrix} 10 & 31 \\ 21 & 19 \end{bmatrix}, B_4 = \begin{bmatrix} 14 & 2 \\ 10 & 25 \end{bmatrix}. \qquad (11)$$

Step 6: The entries of the blocks B_i $(1 \leq j \leq 4)$ in (11) are as follows.

$b_j^1 = 7, 10, 15, 2$ for $j = 1, 2, 3, 4$
$b_j^2 = 3, 16, 4, 4$ for $j = 1, 2, 3, 4$
$b_j^3 = 10, 31,21,19$ for $j = 1, 2, 3, 4$
$b_j^4 = 14, 2, 10, 25$ for $j = 1, 2, 3, 4$

respectively.

Step 7: The d_i determinants of the blocks B_i in (11) are computed as seen below:

$d_1 = \det(B_1) = -136$	$d_3 = \det(B_3) = -461$
$d_2 = \det(B_2) = -52$	$d_4 = \det(B_4) = 330.$

Step 8: Merging the entries of the *E* matrix from the previous steps, the following matrix is obtained:

$$E = \begin{bmatrix} -136 & 7 & 10 & 2 \\ -52 & 3 & 16 & 4 \\ -461 & 10 & 31 & 19 \\ 330 & 14 & 2 & 25 \end{bmatrix}.$$

As a result, here the phase of the encryption is over.

Decryption Algorithm:
Step 1: Start
Step 2: Using the public key $n = 2$, the matrix $Q_h^2(x)$ can be found easily:

$$Q_h^2(x) = \begin{bmatrix} F_{h,3}(x) & F_{h,2}(x) \\ F_{h,2}(x) & F_{h,1}(x) \end{bmatrix}$$

Step 3: From the equation (6), the entries of $Q_h^2(x)$ are obtained as the following step. These entries:

$r_1 = h^2(x) + 1$	$r_2 = h(x)$	$r_3 = h(x)$	$r_4 = 1.$

To construct the matrix *G*, we compute the elements $g_{1,2}^i$ in the Step 4 and Step 5.

Step 4-5: These elements are found by

$$\begin{aligned}
g_1^1 &= 7(h^2(x) + 1) + 10\, h(x),\\
g_1^2 &= 3(h^2(x) + 1) + 16\, h(x),\\
g_1^3 &= 10(h^2(x) + 1) + 31h(x),\\
g_1^4 &= 14(h^2(x) + 1) + 2h(x).
\end{aligned}$$

$$\begin{aligned}
g_2^1 &= 7\, h(x) + 10,\\
g_2^2 &= 3\, h(x) + 16,\\
g_2^3 &= 10\, h(x) + 31,\\
g_2^4 &= 14\, h(x) + 2.
\end{aligned}$$

Step 6-7: The x_i elements for $i = 1, 2, 3, 4$ are calculated and then replaced by x_i with b_3^i:

$(-1)^4(-136) = g_1^1(h(x)\, x_1 + 1.2) + g_2^1\big((h^2(x) + 1)\, x_1 + 2h(x)\big) \Rightarrow$ $x_1 = b_3^1 = 15.$
$(-1)^4(-52) = g_1^2(h(x)\, x_2 + 1.4) + g_2^2\big((h^2(x) + 1)\, x_2 + 4h(x)\big) \Rightarrow$
$x_2 = b_3^2 = 4.$
$(-1)^4(-461) = g_1^3(h(x)\, x_3 + 1.19) + g_2^3\big((h^2(x) + 1)\, x_2 + 19h(x)\big) \Rightarrow$ $x_3 = b_3^3 = 21.$
$(-1)^4(300) = g_1^4(h(x)\, x_4 + 1.25) + g_2^4\big((h^2(x) + 1)\, x_4 + 25h(x)\big) \Rightarrow$ $x_4 = b_3^4 = 10.$

Step 8: The block matrices, B_i are built as shown below:

$$B_1 = \begin{bmatrix} 7 & 10 \\ 15 & 2 \end{bmatrix}, B_2 = \begin{bmatrix} 3 & 16 \\ 4 & 4 \end{bmatrix}, B_3 = \begin{bmatrix} 10 & 31 \\ 21 & 19 \end{bmatrix}, B_4 = \begin{bmatrix} 14 & 2 \\ 10 & 25 \end{bmatrix}.$$

Step 9: By merging these matrices B_1, B_2, B_3, B_4, the following matrix M is found.

$$M = \begin{bmatrix} 7 & 10 & 3 & 16 \\ 15 & 2 & 4 & 4 \\ 10 & 31 & 14 & 2 \\ 21 & 19 & 10 & 25 \end{bmatrix}.$$

Step 10: To convert the entries of the message matrix M to numeric values, using ASCII table in [23], the following initial message is constructed:

$$M = \begin{bmatrix} BELL & LF & ETX & DLE \\ SI & STX & EOT & EOT \\ LF & US & SO & STX \\ NAK & DC3 & LF & EM \end{bmatrix}_{4x4}$$

Step 11: End.

Now, we provide an alternate implementation of this algorithm in the example below.

Example 1.2: Now let's build the message matrix for the given message text: "CODE"

Coding Algorithm:
Step 1: Start
Step 2: Then, we obtain the following message matrix *M* using this message text:

$$M = \begin{bmatrix} C & O \\ D & E \end{bmatrix}_{2x2} \tag{12}$$

Step 3: In this step, we determine that $n = 1$ because $\left[\!\left[b/_2 \right]\!\right] = n$ via the equation (8).
Step 4: By using ASCII decimal in [24], all elements in the *M* message matrix in (12) are transformed.

$$M = \begin{bmatrix} 67 & 79 \\ 68 & 69 \end{bmatrix}_{2x2}$$
(13)

Step 5: Since *M* is a 2x2 message matrix, there is only one block matrix, B_1 as follows:

$$B_1 = \begin{bmatrix} 67 & 79 \\ 68 & 69 \end{bmatrix}_{2x2}. \tag{14}$$

Step 6: The following are the elements of block B_1 (1≤j≤4) in (14**)**

$b_j^1 = 67, 79, 68, 69$ for $j = 1, 2, 3, 4$

Step 7: The following computation shows the d_1 determinant of the block B_1 in (14):

$$d_1 = \det(B_1) = -749$$

Step 8: After combining the E matrix's entries from the earlier stages, the following matrix is produced:

$$E = \begin{bmatrix} -749 & 67 \\ 79 & 69 \end{bmatrix}$$

Decryption Algorithm:
Step 1: Start
Step 2: Using the public key $n = 1$, the matrix $Q_h^1(x)$ can be found easily:

$$Q_h^1(x) = \begin{bmatrix} F_{h,2}(x) & F_{h,1}(x) \\ F_{h,1}(x) & F_{h,0}(x) \end{bmatrix}$$

Step 3: The next step is to derive the entries of $Q_h^2(x)$ from equation (6):

$r_1 = h(x)$	$r_2 = 1$	$r_3 = 1$	$r_4 = 0.$

In Steps 4 and 5, we compute the elements $g_{1,2}^i$ in order to build the matrix G.

Step 4-5: These elements are obtained by

$$g_1^1 = 67\big(h(x)\big) + 79\ .1,$$

$$g_2^1 = 67\ .1 + 79.0.$$

Step 6-7: After computing the x_1 element, x_1 is substituted with b_3^1:

$$\boxed{\begin{gathered}(-1)^1(-749) = g_1^1(1.\ x_1 + 0.69) - g_2^1(h(x)\ x_1 + 69.1) \Rightarrow \\ x_1 = b_3^1 = 68.\end{gathered}}$$

Step 8: The block matrix B_1, which is one, is found as follows:

$$B_1 = \begin{bmatrix} 7 & 10 \\ 15 & 2 \end{bmatrix}.$$

Step 9: Without the need for merging, the *M* message matrix is obtained as follows:

$$M = \begin{bmatrix} 67 & 79 \\ 68 & 69 \end{bmatrix}_{4x4}.$$

Step 10: Similarly, the initial matrix can be easily found with the help of the ASCII table in [23].

$$M = \begin{bmatrix} C & O \\ D & E \end{bmatrix}_{4x4}$$

Step 11: End.

3. Evaluation the performance of the algorithm:

In this section, we shall consider the mathematical operations in the previous section. In this method, *M* is *plain text* which is also referred to as an unencrypted text in cryptography. In addition, by converting *plain text* to *cipher text* using various encryption algorithms, messages are encoded in a way that cannot be read. Here, *E* is *cipher text.* Then, this *cipher text* is converted to *plain text* by decryption algorithms. As in [18], the encryption and decryption time of the whole algorithm are calculated. To compute this duration, the following table is used for all the steps of the encryption and decryption of this method:

Table 1: The details of the name of process and notation of the algorithm

Encryption		**Decryption**	
The Name of Process	*Notation*	*The Name of Process*	*Notation*
Construct the message matrix *M*	$\Delta\ t_{cons}$	Computing the elements of the matrix $Q_h^n(x)$ with public key n.	$\Delta\ t_{com}$
Converting the message matrix *M* to ASCII	$\Delta\ t_{conv}$	Assignment of the entries of the matrix $Q_h^n(x)$	$\Delta\ t_a$
Finding the public key n	$\Delta\ t_f$	Computing g_j^i elements for $j = 1, 2$	$\Delta\ t_{com}$
Splitting the message matrix into B_i blocks	$\Delta\ t_s$	Computing x_i elements for $i = 1, 2$,3,4.	$\Delta\ t_{com}$
Assignment of the entries of the blocks B_i	$\Delta\ t_a$	Construct the block matrices B_i (replace by x_i with b_3^i)	$\Delta\ t_{cons}$
Computing d_i determinants of b_j^i blocks for $i, j = 1, 2, 3, 4$	$\Delta\ t_{com}$	Construct the message matrix *M*	$\Delta\ t_{cons}$
Merging the entries of the *E* matrix	$\Delta\ t_m$	Converting the message matrix ASCII decimal to numeric values	$\Delta\ t_{conv}$

According to the Table 1, we consider all the steps of full encryption time, T_{enc} and decryption time, T_{dec}. These results are found:

$$T_{enc} = \Delta_{t_{cons}} + \Delta_{t_{conv}} + \Delta_{t_f} + \Delta_{t_s} + \Delta_{t_a} + \Delta_{t_{com}} + \Delta_{t_m}, \tag{15}$$

$$T_{dec} = 3\Delta_{t_{com}} + 2\Delta_{t_{cons}} + \Delta_{t_{conv}} + \Delta_{t_a}. \tag{16}$$

Both the encryption and decryption phases of the model from the results (15) and (16) are based on eight basic operational operations.

4. Conclusion:

This algorithm has the following superior features:

(a) As we know, one of the biggest problems is slowness in the encryption methods. To overcome this problem, we use symmetric encryption in this method. When we also examine the speed of the model by reducing it to a transactional basis, we realize that the results from (15) and (16) show that this method is so fast.

(b) The alphabet system in this study is used to the ASCII (American Standard Code for Information Interchange) character code system and thus it is aimed to make it ready for cryptographic applications. In addition, it is difficult to find combinations using brute force attack of plain text with algorithms created based on this coding structure. [26] For this reason, one of the biggest advantages of the algorithm proposed because this coding structure is used is that the infrastructure of this coding structure is suitable for security tests. On the other hand, the method that we have proposed allows us to understand that it can be used for many types of polynomials.

When evaluating the performance of this model, it can be said that the method is suitable for both encryption/decryption time and ASCII character encoding for developing secure cryptographic applications.

References:

[1] Lee, G., Asci, M., Some properties of the (p,q)-Fibonacci and (p,q)-Lucas polynomials, **J. Appl. Math**, 264842, (2012).

[2] Simsek, Y. Construction of general forms of ordinary generating functions for more families of numbers and multiple variables polynomials, **Revista de la Real Academia de Ciencias Exactas, Físicas y Naturales. Serie A. Matemáticas**, 117, 3, 130, (2023).

[3] Zhang, C., Khan, W. A. and Kızılateş, C., On (p,q)–Fibonacci and (p,q)–Lucas Polynomials Associated with Changhee Numbers and Their Properties, **Symmetry**, 15(4), 851, (2023).

[4] Prasad, B., Coding theory on Lucas p -numbers, **Discrete Math. Algorithms Appl.**, 17,8, no.4, 17 pages, (2016).

[5] Basu, M. and Prasad, B., The generalized relations among the code elements for Fibonacci coding theory, **Chaos Solitons Fractals**, 41, 5, 2517-2525, (2009).

[6] Koshy T., **Fibonacci and Lucas Numbers with Applications**, Toronto, New York, NY, USA, (2001).

[7] Nalli, A., Haukkanen, P., On generalized Fibonacci and Lucas polynomials, **Chaos Solitons Fractals,** 42, 5, 3179–3186, (2009).

[8] Catarino, P., A note on $h(x)$–Fibonacci quaternion polynomials. **Chaos Solitons Fractals,** 77, 1–5, (2015).

[9] Birol, F., Koruoğlu, Ö., Linear groups related to Fibonacci polynomials, **Adv. Studies: Euro-Tbilisi Math. J.**, 15, 4, 29-40, (2022).

[10] Wang, W., Wang, H., Generalized-Humbert Polynomials via Generalized Fibonacci Polynomials, **Applied Mathematics and Computation,** 307, 204–216, (2017).

[11] Kızılateş, C., Cekim. B, Tuglu N and Kim T., New families of three-variable polynomials coupled with well-known polynomials and numbers, **Symmetry**, 264, 11, 1-13, (2019).

[12] Taş, N., Uçar, S., Özgür, N. Y. and Kaymak, Ö. Ö., A new coding/decoding algorithm using Fibonacci numbers, **Discrete Math. Algorithms Appl.**, 10, 2, 1850028, (2018).

[13] Stakhov, A. P., Fibonacci matrices, a generalization of the Cassini formula and a new coding theory, **Chaos Solitons Fractals**, 30, 1, 56-66, (2006).

[14] Uçar, S.; Tas, N.; Özgür, N.Y. A new Application to coding Theory via Fibonacci numbers, **Math. Sci. Appl. Notes**, 7, 62–70, (2019).

[15] Basu, M., Prasad, B., Coding theory on the m-extension of the Fibonacci p-numbers, **Chaos, Solitons & Fractals**, 42, 4, 2522-2530, (2009).

[16] Prasad, B., Coding theory on $(h(x), g(y))$-extension of the Fibonacci *p*-numbers polynomials, **Universal Journal of Computational Mathematics,** 2, 1, 6-10, (2014).

[17] Prasad, B., Basu, M., Coding theory on $h(x)$ Fibonacci *p*-numbers polynomials, **Discrete Mathematics, Algorithms and Applications**, 4, 3, 1250030, (2012).

[18] Stakhov, A.P., The ‘‘golden’’ matrices and a new kind of cryptography, **Chaos, Solitons & Fractals**, 32,1138–1146, (2007).

[19] Akbiyik, M., Alo, J., On Third-Order Bronze Fibonacci Numbers, **Mathematics**, 9, 20:2606, (2021).

[20] Aydınyuz, S., Asçi, M., Error detection and correction for coding theory on k -order Gaussian Fibonacci matrices, **Mathematical Biosciences and Engineering**, 20(2): 1993–2010, (2022).

[21] Asci, M., Aydınyuz, S., k -order Gaussian Fibonacci polynomials and applications to the coding/decoding theory, **Journal of Discrete Mathematical Sciences and Cryptography**, 25, 5, 1399-1416, (2022).

[22] Basu, M., Das, M., Coding theory on generalized Fibonacci *n*-step polynomials, **Journal of Information and Optimization Sciences**, 38, 1, (2017).

[23] ASCII, https://en.wikipedia.org/wiki/ASCII, (14.08.2023).

[24] Newline, https://en.wikipedia.org/wiki/Newline, (19.07.2023).

[25] Asci, M., Aydinyuz S., *k*-Order Fibonacci Polynomials on AES-Like Cryptology, **CMES-Computer Modeling in Engineering & Sciences**, 131(1), 277–293, (2022).

[26] Shaik, A., ASCII Binary Self Generated Key Encryption, **The International Journal of Computer Science & Applications (TIJCSA),** 1(3), 112-115, (2012).